第十二章 并发程序的测试（云里雾里）

1. 并发测试

（1）安全测试

通常采用不变性条件的形式，即判断某个类的行为是否与其规范保持一致。

（2）活跃性测试

（a）进展测试

（b）无进展测试

（c）与活跃性相关的是性能测试

衡量性能的指标：

吞吐量、响应性、可伸缩性

#### 一．正确性测试

1. 在为某个并发类设计测试单元时，首先需要执行与测试串行类时相同的分析----找出检查的不变性条件和后验条件。

2. 安全性测试

影子列表、校验和

3. 资源管理的测试

了解各种资源管理工具

4. 使用回调

#### 二．性能测试

1. 吞吐量衡量

2. 响应性衡量

公平模式的信号量能实现更低的变动性

非公平模式的信号量能实现更好的吞吐量

线程的数量

#### 三．避免性能测试的陷阱

1. 垃圾回收

我们无法预测垃圾回收的执行顺序，因此垃圾回收可能在测试时的任何时刻运行，为了防止测试结果产生偏差，有两种策略。

（1）确保垃圾回收在整个期间都不会执行（再调用JVM时指定-verbose）。

（2）确保垃圾回收操作在测试期间执行多次，这样测试程序就能充分反映出运行期间的内存分配与垃圾回收等开销。

2. 动态编译

（1）解释执行：

当某个类第一次被加载时，JVM会通过解译字节码的方法来执行它。而且基于各种原因，代码还可能会反编译，即从直接执行变为解释执行。

（2）直接执行：

在某个时刻，如果一个方法运行的次数足够多，那么动态编译器，会将它编译为机器代码，代码的执行方式将变为直接执行。

有两种解决办法

（1）使程序运行足够长的时间

（2）使代码预先运行一段时间

在运行程序时使用命令行选项-xx:+PrintCompilation会在编译完成后打印一条信息

3.对代码路径的不真实采样

动态编译可能会让不同地方调用的同一方法生成的代码不同

应该将单线程的性能测试与多线程的性能测试结合在一起。

4. 不真实的竞争程度

5. 无用的代码消除

优化编译器能找出并消除那些不会对输出结果产生任何影响的无用代码。

建议使用-server模式而不是-client模式。

#### 四．其他的测试方法

1. 代码审查

2. 静态分析工具

例如FindBugs等。

3. 面向方面的测试技术

4. 分析与检测工具